A Programmable Network Based Approach for Managing Dynamic Virtual Private Networks

Radu State, Olivier Festor, Emmanuel Nataf
LORIA - INRIA Lorraine - Université de Nancy II
615 rue du Jardin Botanique
F-54602 Villes-les-Nancy Cedex
France

Abstract *In this paper we address the management of dynamic virtual private networks (DVPN). Dynamic virtual private networks are VPNs with a high degree of change in terms of membership, implying the necessity for fast reconfiguration and provisioning. We propose a framework for the management of such networks by integrating the management and the control plane, using the programmable and active networks paradigms developed within the research community. We apply the framework to the management of residential user TV multicast, where an ATM based access network supports the delivery of TV content to all clients having subscribed to a DVPN.*

Keywords: Active Network, P1520, VPN, multicast

1 Introduction

The advent of broadband technologies for the local loop, combined with the deregulation of this part of the network in most european countries, fosters the deployment of new services to the end user. One of these services is the multicasting of digital TV channels to all subscribed residential customers. Such a service requires both a dedicated signalling plane (e.g. for channel selection by the end-user) and a high performance management plane for provisionning, monitoring and flow management.

The project aims at providing a management framework for dynamic virtual private networks (DVPN) in the backbone (here a metropolitan area network) to provision the local loop. Each TV channel is conceptually defined as a multicast tree which has the caracteristics of a VPN, ie. Closed User Group, security and QoS guarantees. The support of multicast facility within VPNs is required in order to optimize the use of network resources. Since the required multicast trees are strongly dynamic, their configuration within the traditional VPN management time-scale is no more viable. Our work proposes an integration of the management and signalling planes using programmable and active technologies in order to cope with this strong variability and dynamics.

To present the major components of our architecture, the remainder of the paper is organized as follows. Section 2 gives a definition of Dynamic Virtual Private Networks, illustrates the target backbone to be managed and motivates the need for a programmable architecture in order to enable in-time management. Section 3 and 4 detail the components of our management architecture. Section 5 summarizes the work done in other projects which has been partially reused in our approach. Finally a short conclusion is given and future work is outlined.

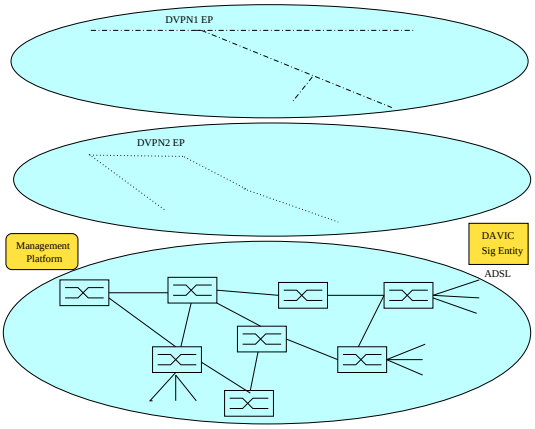

FIG. 1 – *The network components*

2 Dynamic Virtual Private Networks

Dynamic Virtual Private Networks are from a logical point of view virtual private networks which encapsulate multicast trees whose topology may change very frequently depending on user interactions (join, leave a channel). These DVPN rely on Service Level Agreements (SLA) established between individual end-user subscribers and the service provider as well as on SLAs between the latter and content providers. The service provider may itself rely on a transport provider with specific SLAs.

Figure 1 illustrates the different components of the network to be managed. At the lowest level, the network is composed of ATM switches. At the backbone edges, customers access the dynamic virtual network through Service Access Points (SAP). The physical infrastructure uses the Asymmetric Digital Subscriber Line (ADSL) technology. Using a Set-Top-Box, a user can zap from one channel to another through DAVIC [1]like signalling. This selection is issued to a Customer signalling entity which transforms this query into a tree expansion request to the DVPN management entity. The latter is responsible for configuring the network to deliver the flow of the selected channel to the appropriate end-user.

Each TV channel is modeled as a DVPN. This choice is appropriate since common features which are particular to VPNs (e.g; Closed User Group, Security and QoS guarantees) are of a crucial importance in the context of TV residential broadcast. At the network level, it represents a multicast tree as illustrated through the two DVPNs in figure 1.

Traditional Management time-scale for VPN provision is not appropriate for such DVPNs. In fact, the topology change of each tree must occur in a couple of milliseconds whenever a customer zaps from one channel to another. Moreover, this configuration task must be performed in parallel enabling multiple customers to change channels simultaneously. Since generic network layer support is unlikely to support such requirements, part of the management tasks must be integrated with the signalling facility into a programmable infrastructure. Time constraints are one motivation, information sharing provides a second one. Both camps benefit: like in active networking, signalling protocols may benefit from information provided by the management framework (like topology or link states), and on the other hand, a management framework may benefit from information provided by the value added services signalling plane (e.g. actual number of customers who are looking at a given channel).

3 The management architecture

In order to enable management of those DVPNs, we have chosen to combine the programmable network approach and active technology, as detailled in this section.

As illustrated in figure 2, the management architecture is built within a CORBA DPE on top of P1520 [3, 2] abstract switch interfaces. The entire architecture is composed of five elements.

The main component is the DVPN Tree Manager. This entity is responsible for the configuration, extension and reduction of DVPN trees on the backbone. It maintains a view of the topology of the physical network as well as a logical one for each each DVPN currently in activity. This entity offers an API to the Custo-

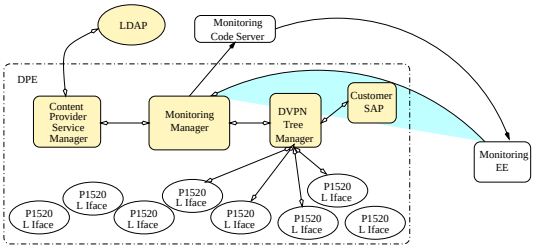

FIG. 2 – *The management architecture building blocks*

mer Service Access Point (second component) through which all channel setup/change requests are received, acknowledged (check that the user is allowed to join a given DVPN) and performed (issue an expansion request to the DVPN for the given SAP and the given channel.

The third component is a monitoring entity. This management entity provides facilities for deployment of monitoring code for both the Service Provider and the Content Providers. Related to the monitoring entity is the Monitoring EE, an Execution Environment for active code. This EE is avalaible in various nodes of the network, especially on each edge node as one special End-User. This EE is used by the Monitoring manager to deploy service specific monitoring code for one or multiple content provider service parameters. This monitoring code is developped by the network management staff according to SLAs. The resulting code can be dynamically downloaded and controlled by the Monitoring Manager in accordance with the service management entity described below. The functionality that we achieve with this approach corresponds to a on-the-fly construction of RMON Mibs. For usual trafic monitoring, the monitoring manager uses SNMP-based agents located in the switches, at least in the first release of the management environment. The monitoring manager relies on a monitoring code server offering a set of monitoring functions which can be deployed to the probe EEs onto the network.

The last component is a service level manager called Content Provider Service Manager, responsible for both DVPN setup and collection of data that must be made available to the content provider (mainly based on what is specified in the Service Level Agreement. This element mainly relies on the monitoring manager to gather information from the network. A second information source is the Customer SAP through which coverage related data can be obtained. The service level reports are made avalaible to the content provider through an LDAP directory server.

All those components are currently under development using the Java technology and a CORBA DPE at the programmable level. The Execution Environment and active code is an extension of the ANTS Toolkit [13] running on a Linux box for probes.

4 DVPN Tree Manager

The DVPN Tree Manager is the core of the management platform.We will first describe the static information model, that is the structure of the information needed to maintain a view on the DVPN and public network properties. Afterwards we will continue with the functional aspects of its activity.

4.1 Information Model

The information needed in order to perform the management is twofold, since two different layers of abstraction can be put into evidence. The first one concerns the public network used to deliver the DVPN service. The underlying public network is modeled using a slightly modified information model of the one introduced in [12]. The extensions to that model concern the support of multicast connections at a layer trail and respectively subnetwork connection layer. At the DVPN level, new entities are introduced and linked to the supporting elements from the public network layer view (see figure 3). For the sake of clarity we keep the public network part of the information model quite simple and generic. Specific ATM related entities are obtained by a specialization of the generic classes. For instance :

– the `LayerTrail` class can be derived in or-

der to stand for a ATM SVC;

- By a specialization of the `Subnetwork` class, one can model physical ATM switches. However, this class can also model a collection of ATM switches performing as one global switch.
- `SNC` (Subnetwork Connection) represents physical crossconnects done in the switches, but also communication across a collection of switches, considered to jointly form a `Subnetwork` entity. It is worth noting that the `SNC` object has several destination endpoints and one designated source endpoint in order to perform at switch level the point to multipoint crossconnections.
- `nwCTP` models connection endpoints. In the case of a ATM residential backbone, it is mapped to a triplet (port, VPI, VCI) representing the interface and the connection identifiers for the particular connection.
- `Link` represents the connectivity at the physical layer.
- `LinkTP` models the termination point of a `Link` object and is mapped to the particular switch interface used.

The information model for VPNs, that we extended for reuse in the context of DVPN, has been introduced in [9]. We enhance the model in order to be able to:

- model a collection of DVPNs as seen by a global view. The class VPNForest is introduced for this purpose.
- represent the Multicast Tree that corresponds to a particular DVPN. At the DVPN level, such a tree is a collection of network flow endpoints (`NFEP`) and the traffic from one source `NFEP` to the remaining ones. This traffic is modeled using the network flow connection class (`NFC`) and represents the video data delivered to the end users. The `NFC` is a one-to-multipoint connection in order to model a multicast tree. It is characterized by a series of QoS parameters. If MPEG format is used, these parameters can be for instance, the I, B and respectively P frame rates. The active code based monitoring is performed on this type of objects. The content provider is able to check if SLAs parameters are satisfied by deploying application specific code. For instance, it can simulate client behaviour and measure the performance of the provided service.
- A DVPN member corresponds to a a network address, used by a particular residential user, and that is subscribed to DVPNs.

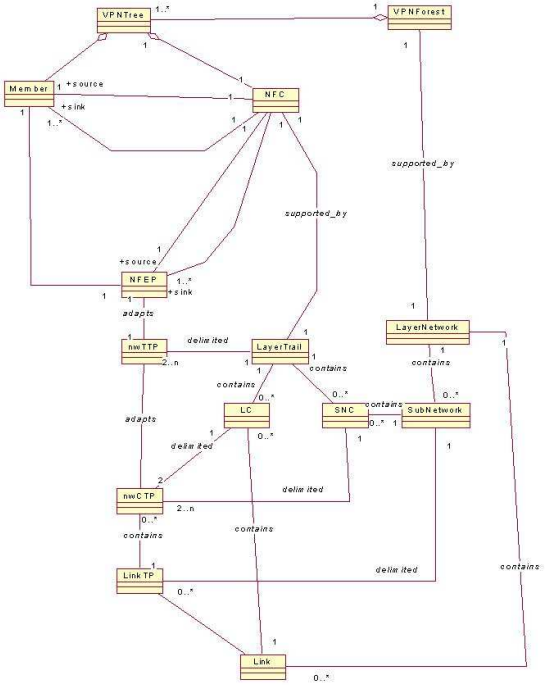

FIG. 3 – *DVPN Information Model*

The dynamic reconfiguration of the tree is initated at this level first and down-propagated to the network level. Several causes can trigger such a reconfiguration.

- users that join a DVPN,
- users leaving a DVPN,
- Management initiated actions. For instance, stationary parts of the multicast tree, can be merged onto a one-to-many permanent connection in order to facilitate management.

For instance, if one user desires to join a DVPN, its SAP asks the DVPN Tree manager to add a branch to the particular `NFC`. At the network level, this operation is translated in adding a branch to the trail supporting the `NFC`. Since a `LayerTrail` is build from interconnected `SNC` and `LC`, these objects must be created/modified. The usual operation flow involves the addition of branch to an existent `SNC` object that is already used to multicast the DVPN traffic, and the necessary `LC` and `SNC` objects are created in order to deliver it to the users's access point. In case of users leaving a DVPN, there might be the case that whole areas of the multicast tree need to be teared down in order to release bandwidth resources.

4.2 The computational objects

We will show in this section a functional representation of the DVPN Tree Manager in terms of computational objects and related interfaces. The figure 4 illustrates computational objects of the DVPN Tree Manager and those that are related with them.

- Route Manager will mainly compute efficient tree expansion needed for a particular user to join a DVPN. It will use the instantiated information model and existent Multicast tree building algorithms, like the one introduced in [5], in order to achieve this goal.
- Tear-Down Manager frees up the resources if necessary. This is the case, when a particular multicast subtree is no longer needed over a particular area of the network. It performs at a different time scale than the Route Manager, since connection set up has to be fast compared to connection removal, which itself can be done on a longer time scale. This approach is also motivated by usual customer behavior, where zapping during advertising breaks in a long term watched channel is performed.
- Connection Manager performs the necessary connection set-ups, or removals, on behalf of the Route Manager or Tear-Down Manager. For such a request, it will access the network switches using the provided P1520 interface in order to create/delete necessary connections.
- Resource Manager monitors the status of network resources in current use. It gathers the required information from the Connection Manager and provides sufficient information to the Route and Tear-Down Managers. For instance, in case of a link going down the Route Manager is informed in order to update its network view.
- Client represents the immediate service access available to one Client SAP. It receives client signalisation (DAVIC type like) and transforms it to requests invoked on the Route and Tear-Down Managers.
- Access Manager allows client actions with respect to established authentification/access rights and service level agreements.

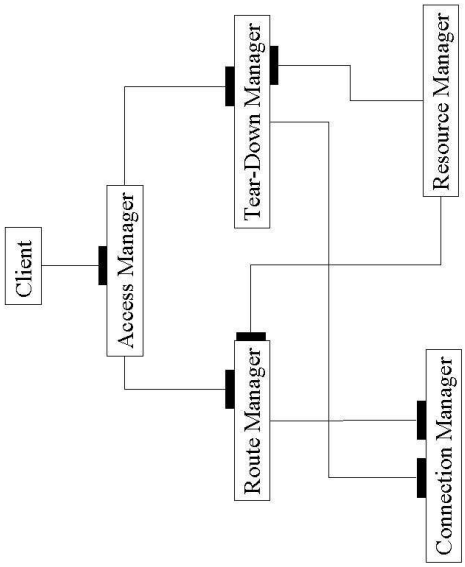

FIG. 4 – *DVPN Computational Objects*

5 Related work

In [8] a special VPN architecture, called "the hose model" has been introduced, providing for efficient resource utilization and enhanced client flexibility. A VPN is defined in terms of aggregate traffic from one endpoint in the VPN to all other endpoints of the VPN. Our DVPN is a special type of a dynamic "hose", where the same amount of traffic from the source is delivered to all endpoints using the multicasting technology. From the architectural framework point of view, our approach relies on the APIs defined in the IEEE PIN project and at the Columbia University. Currently we base our specifications on the ATM Switch Resource Abstractions APIs. Concerning VPNs, the Genesis project [6] proposes a distributed network operating system based on those APIs as well as a spawning architecture for their setup. Their objectives are quite different from the ones defined in our framework. The Genesis kernel could be used in our framework to spawn an initial DVPN. The idea of combining programmable and active technology has already been proposed in the mobiware framework [7, 4], where the programmable framework was used to build the signalling plane, and the active technology permitted to inject code to adapt the data plane to wireless customer-tailored specific QoS conditions. The principle developped in our approach is conceptually similar but applied to the management plane for the active technology part. The use of active technology for management has been proposed several times over the last year. Most of these approaches deal with standard delegation, mostly in conjunction with a legacy SNMP environment. This is done for instance in the SmartPackets approach [11] where active packets are used to delegate monitoring and access to SNMP variables. Our approach for the active technology part is different from those approaches in the way that the active code sent to monitoring Execution Environments is not dependent on standard SNMP, but provides application specific flow monitoring facilities to instrument the edge probes. This is similar to what is proposed in ANMAC [10].

6 Conclusion and future work

In this paper, we have presented a framework which combines both a programmable paradigm and the use of active technology for the monitoring of dynamic virtual private networks. The use of a programmable network architecture is motivated by the need to combine both application level signalling and network level management.

At the programmable network level we have implemented a DVPN model and a prototype configuration manager. This manager is fully integrated in the DPE, enabling thus performant information exchange with the signalling facility.

Using active technology for dynamic monitoring represents in our framework much more than just management by delegation. In fact, using this technology enables deployment of management functions that are adapted to a given type of application, gathering the semantics of the data flows. This is very interesting for making service management more efficient and user-friendly.

The current approach is limited to multicast trees based on a single source for each VPN and multiple data sinks. One future direction of this work will be to extend the approach to multi-source multicast groups. Future work will also investigate the dynamic decomposition/composition of VPNs, in order to model dynamic interactions among enterprise networks. This could be the case for instance, when one enterprise VPN is split into several VPNs, or several VPNs must be joined in order to permit a common work of the involved users in one VPN over a delimited period of time.

Références

[1] Description of DAVIC Functionalities. *Digital Audio-Visual Consortium*, 1996.

[2] C.M. Adam, A.A. Lazar, and M. Nandikesan. ATM Switch Resource Abstractions,

March 1999. IEEE/WG P1520/TS/ATM-017 Working Document.

[3] C.M. Adam, A.A. Lazar, and M. Nandikesan. Switch abstractions for designing open interfaces, March 1999. IEEE/WG P1520/TS/ATM-016 Working Document.

[4] O. Angin, A.T. Campbell, M.E. Kounavis, and R.F. Liao. The Mobiware Toolkit: Programmable Support for Adaptive Mobile Networking. *IEEE Personal Communications Mag., Special Issue on Adapting to Network and Client Variability*, 5(4):32–44, August 1998.

[5] F. Bauer and A.Varma. ARIES: A rearrangeable inexpensive edge-based on-line Steiner algorithm. *IEEE Journal on Selected Areas in Communication*, 15(3):382–397, April 1998.

[6] A. Campbell, H.G. De Meer, M.E. Kounavis, K. Miki, J. Vicente, and D.A. Villela. The Genesis Kernel: A Virtual Network Operating System for Spawning Network Architectures. In *2nd Int'l Conf. on Open Architectures and Network Programming*, N.Y., May 1999.

[7] A.T. Campbell, M.E. Kounavis, and R.F. Liao. Programmable Mobile Networks. *Computer Networks and ISDN Systems, Computer Networks*, 31, April 1999.

[8] N.G. Duffield, P. Goyal, A.Greenberg, P. Mishra, K.K. Ramakrishnan, and J.E. van der Merwe. A flexible model for resource management in virtual private networks. In *SIGCOMM'99*, pages 95–107. ACM, 1999.

[9] E.C. Kim, C.S. Hong, and J.G. Song. The multi-layer vpn management architecture. In *Proc. NOMS 1999*, pages 187–199, 1999.

[10] S. Norden and K. Wong. ANMAC: An Architectural Framework for Network Management and Control Using Active Networks. In S. Covaci, editor, *Active Networks: Proc. First International Working Conference, IWAN'99*, pages 212–219, Berlin, Germany, June 1999. Springer Verlag, LNCS 1653.

[11] B. Schwartz, W. Zhou, A.W. Jackson, W.T. Strayer, D. Rockwell, and C. Partridge. Smart Packets for Active Networks. In *Proc. OpenArch '99*, March 1999.

[12] R. State, E. Nataf, and O. Festor. Poster session: A Java based Implementation of a Network Level Information Model for the ATM/Frame Relay Interconnection. In *Proc. NOMS 2000*, April 2000.

[13] D.J. Wetherall, J.V. Guttag, and D.L. Tennenhouse. ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols,. In *Proc. IEEE OpenArch'98*, 1998.